

ISIS-II PL/M-86 V2.0 COMPILATION OF MODULE CM1
 OBJECT MODULE PLACED IN CM.OBJ
 COMPILER INVOKED BY: :F0: CM.PLX DEBUG OPTIMIZE(2) DATE(FEBRUARY 2, 1982) XREF

```
1      %title ('codemacro module 1')
      cm1:
      do;

/*
```

modified 7/24/81 R. Silberstein

```
%/
```

```
/* This is the module to build new instructions
   which is not present in the already existing
   system. */
```

```
%include (:f1:macro.lit)
= %nolist
%include (:f1:equals.lit)
= %nolist
%include (:f1:struct.lit)
= %nolist
%include (:f1:cmacd.lit)
= %nolist
%include (:f1:termmod.lit)
= %nolist
%include (:f1:scan.ext)
= %nolist
%include (:f1:termmod.ext)
= %nolist
%include (:f1:cm2.ext)
= %nolist
%include (:f1:cm.lit)
= %nolist
%include (:f1:global.ext)
= %nolist
```

CP/M-86 v.1.1 (vol. IV)

COPYRIGHT © 1981, 1982

DIGITAL RESEARCH

P. O. BOX 579

PACIFIC GROVE, CA 93950

SER. # C81-162

```
$object
```

```
/* Subroutines: */
```

```

59 1  more$left$on$line: PROC byte;
60 2    if accum(0) <> cr then return true;
62 2                                else return false;
63 2  end$proc more$left$on$line;

64 1  modrm$rout: PROC;
65 2  dcl nopar byte;
66 2    if token.type = number then$do
68 3      call put$b(mmodrm1);
69 3      if token.value > 7 then$do
71 4        cm$error=true; /* legal values are ^-1,... ,7 */
72 4        return;
73 4      elseif$do
75 4        call put$b(token.value);
76 4      end$if;
77 3    elseif$do
79 3      if token.type = ident then$do
81 4        if legal$parameter(acclen,.accum(0),.nopar) then$do
83 5          call put$b(mmodrm2);
84 5          call put$b(nopar);
85 5        elseif$do /* error, parameter mismatch */
87 5          cm$error=true;
88 5          return;
89 5        end$if;
90 4      elseif$do /* error, expected parameter */
92 4        cm$error=true;
93 4        return;
94 4      end$if;
95 3    end$if;
96 2    call scan;
97 2    if accum(0) <> comma then$do
    /* error, expected comma */
99 3      cm$error=true;
100 3      return;
101 3    end$if;
102 2    call scan;
103 2    if token.type = ident then$do
105 3      if legal$parameter(acclen,.accum(0),.nopar) then$do
107 4        call put$b(nopar);
108 4        call scan;
109 4        return;
110 4      elseif$do /* error, parameter mismatch */
112 4        cm$error=true;
113 4        return;
114 4      end$if;
115 3      elseif$do /* error, expected parameter */
117 3        cm$error=true;
118 3        return;
119 3      end$if;
120 2    end$proc modrm$rout;

121 1  db$dwl$common$rout: PROC(directive);
122 2    dcl (directive,nopar) byte;

```

COPYRIGHT © 1981, 1982

DIGITAL RESEARCH

P. O. BOX 579

PACIFIC GROVE, CA 93950

SER. # _____

```

123 2    if token.type = number then$do
125 3        call put$b(directive);
126 3        if directive = adwn then$do
128 4            call put$w(token.value);
129 4        else$do
131 4            if token.value > 0ffh then cm$error=true;
133 4                else call put$b(token.value);
134 4        endif;
135 3        call scan;
136 3        return;
137 3    else$do
139 3        if token.type = ident then$do
141 4            if legal$parameter(acclen,,accum(0),,nopar) then$do
143 5                call put$b(directive+1);
144 5                call put$b(nopar);
145 5                call scan;
146 5                return;
147 5            else$do /* error, parameter mismatch */
149 5                cm$error=true;
150 5                return;
151 5            endif;
152 4        else$do /* error, expected parameter */
154 4            cm$error=true;
155 4            return;
156 4        endif;
157 3    endif;
158 2    end$proc db$dw$common$rout;

```

```

159 1    d$$rb$rw$rout: PROC(directive);
160 2    dcl (directive,nopar) byte;
161 2    if token.type = ident then$do
163 3        if legal$parameter(acclen,,accum(0),,nopar) then$do
165 4            call put$b(directive);
166 4            call put$b(nopar);
167 4            call scan;
168 4            return;
169 4        else$do /* error, parameter mismatch */
171 4            cm$error=true;
172 4            return;
173 4        endif;
174 3    else$do /* error, expected parameter */
176 3        cm$error=true;
177 3        return;
178 3    endif;
179 2    end$proc d$$rb$rw$rout;

```

```

180 1    nose$fix$rout: PROC;
181 2    dcl nopar byte;
182 2    call put$b(mnose$fix);
183 2    do case legal$seg$reg;
184 3        do; /* error, no segment register specified */
185 4            cm$error=true;
186 4            return;
187 4        end;
188 3        call put$b(res);
189 3        call put$b(rcs);
190 3        call put$b(rss);

```

COPYRIGHT © 1981, 1982
 DIGITAL RESEARCH
 P. O. BOX 575
 PACIFIC GROVE, CA 93950
 SER. # _____

```
191 3      call put$b(rds);
192 3      end$case;
193 2      call scan;
194 2      if accum(0) <> comma then$do
          /* error, expected comma */
196 3      cm$error=true;
197 3      return;
198 3      end$if;
199 2      call scan; /* skip comma */
200 2      if legal$parameter(acclen,.accum(0),.nopar) then$do
202 3      call put$b(nopar);
203 3      call scan;
204 3      else$do
206 3      cm$error=true;
207 3      return;
208 3      end$if;
209 2      end$proc nose$fix$rout;

210 1      dbit$rout: PROC;
211 2      call put$b(mdbit);
212 2      end$proc dbit$rout;

213 1      field$descr$rout: PROC;
214 2      dcl nopar byte,cm$b$var based capt byte;
215 2      do forever;
216 3      if token.type <> number or token.value > 0fh then$do
          /* error, expected numberdef. */
218 4      cm$error=true;
219 4      return;
220 4      end$if;
221 3      call put$b(anumberbits);
222 3      call put$b(token.value);
223 3      call scan;
224 3      if accum(0) <> leftpar then$do
          /* error, expected left parenthesis */
226 4      cm$error=true;
227 4      return;
228 4      end$if;
229 3      call scan; /* skip left parenthesis */
230 3      if token.type = ident then$do
232 4      if not legal$parameter(acclen,.accum(0),.nopar) then$do
          /* error, parameter mismatch */
234 5      cm$error=true;
235 5      return;
236 5      end$if;
237 4      cmpt=cmpt-2;
238 4      cm$b$var=cm$b$var-1; /* it was a parameter, not a number */
239 4      cmpt=cmpt+2;
240 4      call put$b(nopar);
241 4      call scan;
242 4      if accum(0) <> leftpar then$do
          /* error, expected left parenthesis */
244 5      cm$error=true;
245 5      return;
246 5      end$if;
247 4      call scan; /* skip left parenthesis */
248 4      end$if;
```

COPYRIGHT © 1981, 1982

DIGITAL RESEARCH

P. O. BOX 579

PACIFIC GROVE, CA 93950

SER. # _____

```
249 3      if token.type <> number or token.value > 0ffh then$do
          /* error, expected numberdef.(byte) or parameter */
251 4          cm$error=true;
252 4          return;
253 4      end$if;
254 3      call put$b(token.value);
255 3      call scan;
256 3      if accum(0) <> rightpar then$do
          /* error, expected right paranthesis */
258 4          cm$error=true;
259 4          return;
260 4      end$if;
261 3      call scan; /* skip right paranthesis */
262 3      cmpt=cmpt-4;
263 3      if cm$b$var = aformalbits and
          accum(0) = rightpar then call scan;
265 3      cmpt=cmpt+4;
266 3      if accum(0) <> comma then return;
268 3      call scan;
269 3      end$forever;
270 2      end$proc field$descr$rout;

271 1      enddbit$rout: PROC;
272 2          call put$b(mendbit);
273 2      end$proc enddbit$rout;

274 1      endm$rout: PROC;
275 2          call put$b(mendm);
276 2      end$proc endm$rout;
```

COPYRIGHT © 1981, 1982

DIGITAL RESEARCH

P. O. BOX 579

PACIFIC GROVE, CA 93950

SER. # _____

```
%object
```

```
/* level 1 in the syntax-tree of codemacrobuilding */
```

```

277 1  COMMON%CM%ROUT: PROC (TYPE);
278 2  DECLARE TYPE BYTE;
279 2  if pass = 1 then%do
281 3      call skip%rest%of%line;
282 3      return;
283 3  endif;
284 2  cm$error=false;
285 2  if not codemacro%flag then%do
    /* error, codemacro directive outside codemacrobodydef. */
287 3      cm$error=true;
288 3  endif;
289 2  DO CASE TYPE;
290 3      call db%dw%common%rout(adbn);
291 3      call db%dw%common%rout(adwn);
292 3      call d%$rb%rw%rout(mddf);
293 3      call d%$rb%rw%rout(mseqfix);
294 3      call nseqfix%rout;
295 3      call modrm%rout;
296 3      call d%$rb%rw%rout(mrelb);
297 3      call d%$rb%rw%rout(mrelw);
298 3      DO;
299 4          call dbit%rout;
300 4          call field%descr%rout;
301 4          call enddbit%rout;
302 4      END;
303 3  END%CASE;
304 2  if cm$error or more%left%on%line then%do
    /* error */
306 3      global%cm$error=true;
307 3      call errmsg(codemacroerr);
308 3  endif;
309 2  call skip%rest%of%line;
310 2  END COMMON%CM%ROUT;

311 1  codemacro%rout: PROC public;
312 2  if pass = 1 then%do
314 3      codemacro%flag=true;
315 3      call skip%rest%of%line;
316 3      return;
317 3  endif;
318 2  cm$error=false;
319 2  global%cm$error=false;
320 2  if codemacro%flag then%do
    /* error, nested codemacrodefinition */
322 3      cm$error=true;
323 3  endif;
324 2  call init%cm%rout; /* clearing all temp. working storages */
325 2  codemacro%flag=true;
326 2  if not name%rout then%do
    /* error, expected codemacroname */
328 3      cm$error=true;
329 3  endif;
330 2  call formal%list%rout;

```

COPYRIGHT © 1981, 1982
 DIGITAL RESEARCH
 P. O. BOX 579
 PACIFIC GROVE, CA 93950
 SER. # _____

```
331 2      if cm$error or more$left$on$line$ then$do
          /* error */
333 3          global$cm$error=true;
334 3          call errmsg(codemacroerr);
335 3          end$if;
336 2          call skip$rest$of$line;
337 2      end$proc codemacro$rout;

338 1      db$cm$rout: PROC public;
339 2          CALL COMMON$CM$ROUT (0);
340 2      end$proc db$cm$rout;

341 1      dw$cm$rout: PROC public;
342 2          CALL COMMON$CM$ROUT (1);
343 2      end$proc dw$cm$rout;

344 1      dd$cm$rout: PROC public;
345 2          CALL COMMON$CM$ROUT (2);
346 2      end$proc dd$cm$rout;

347 1      segfix$cm$rout: PROC public;
348 2          CALL COMMON$CM$ROUT (3);
349 2      end$proc segfix$cm$rout;

350 1      nose$cm$rout: PROC public;
351 2          CALL COMMON$CM$ROUT (4);
352 2      end$proc nose$cm$rout;

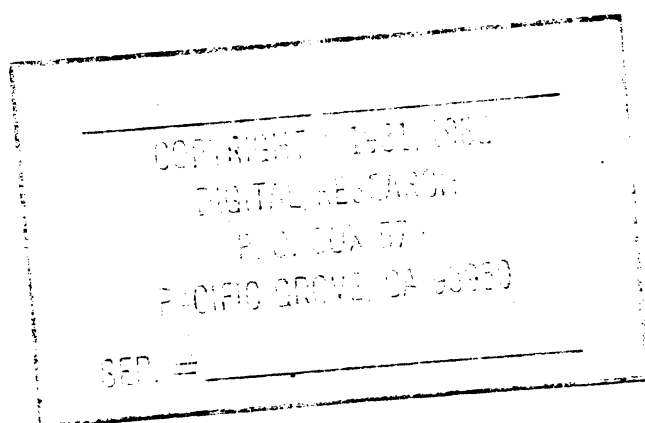
353 1      modrm$cm$rout: PROC public;
354 2          CALL COMMON$CM$ROUT (5);
355 2      end$proc modrm$cm$rout;

356 1      relb$cm$rout: PROC public;
357 2          CALL COMMON$CM$ROUT (6);
358 2      end$proc relb$cm$rout;

359 1      relw$cm$rout: PROC public;
360 2          CALL COMMON$CM$ROUT (7);
361 2      end$proc relw$cm$rout;

362 1      dbit$cm$rout: PROC public;
363 2          CALL COMMON$CM$ROUT (8);
364 2      end$proc dbit$cm$rout;

365 1      end$cm$rout: PROC public;
366 2          if pass = 1 then$do
368 3              call skip$rest$of$line;
369 3              codemacro$flag=false;
370 3              return;
371 3          end$if;
372 2          cm$error=false;
373 2          if not codemacro$flag then$do
          /* error, terminating a not yet started codemacro */
375 3              cm$error=true;
376 3          end$if;
377 2          call endm$rout;
378 2          if pass = 0 then call terminate$cm$rout;
```



```

380 2      if cm$error or more$left$on$line or cm$list$overflow then$do
           /* error */
           call errmsg(codemacroerr);
382 3      endif;
383 3      call skip$rest$of$line;
384 2      global$cm$error=false;
385 2      codemacro$flag=false;
386 2      end$proc end$cm$rout;
387 2
388 1      end$module cm1;

```

COPYRIGHT © 1981, 1982

DIGITAL RESEARCH

P. O. BOX 579

PACIFIC GROVE, CA 93950

SER. # _____

CROSS-REFERENCE LISTING

DEFN	ADDR	SIZE	NAME, ATTRIBUTES, AND REFERENCES
56	0000H	4	ABSADDR. BYTE ARRAY(4) EXTERNAL(51)
56	0000H	1	ACCLEN BYTE EXTERNAL(55) 81 105 141 163 200 232
56	0000H	1	ACCLENSAVE BYTE EXTERNAL(58)
56	0000H	80	ACCUR. BYTE ARRAY(80) EXTERNAL(56) 60 81 97 105 141 163 194 200 224 232 242 256 263 266
56	0000H	80	ACCUHSAVE. BYTE ARRAY(80) EXTERNAL(57)
2			ADDR LITERALLY 56
49	0000H	1	B. BYTE PARAMETER 50
56	0008H	1	BASEINDEX. BYTE MEMBER(OPERANDS)
56	0008H	1	BASEINDEX. BYTE MEMBER(CURRENTSYMBOL)
8			BASEREGBIT LITERALLY
8			BASEREGCOUNT LITERALLY
8			BREGBIT. LITERALLY
8			BREGCOUNT. LITERALLY
4			BYT. LITERALLY
56	0000H	2	CIP. WORD EXTERNAL(20)
1	0000H		CM1. PROCEDURE STACK=0000H
214	0000H	1	CM8VAR BYTE BASED(CMPT) 238 263
35	0000H	1	CMERROR. BYTE EXTERNAL(7) 71 87 92 99 112 117 132 149 154 171 176 185 196 206 218 226 234 244 251 258 284 287 304 318 322 328 331 372 375 380
35	0000H	1	CMLISTOVERFLOW BYTE EXTERNAL(9) 380
35	0000H	2	CMPT WORD EXTERNAL(10) 214 237 238 239 262 263 265
3			CODE LITERALLY
17			CODEMACROERR LITERALLY 307 334 382
56	0000H	1	CODEMACROFLAG. BYTE EXTERNAL(77) 285 314 320 325 369 373 386
8			CODEMACROHEAD. LITERALLY
56	0000H	2	CODEMACROPTR WORD EXTERNAL(72)
311	0313H	91	CODEMACROROUT. PROCEDURE PUBLIC STACK=0006H
55			COLON. LITERALLY
55			COMMA. LITERALLY 97 194 266
277	0286H	141	COMMONCHROUT PROCEDURE STACK=0012H 339 342 345 348 351 354 357 360 363
2			CR LITERALLY 60
56	0000H	2	CSCIP. WORD EXTERNAL(25)
56	0000H	1	CSEGSPEC BYTE EXTERNAL(23)
56	0000H	1	CSEGTTYPE BYTE EXTERNAL(21)
56	0000H	2	CSEGVVALUE. WORD EXTERNAL(22)
12			CSTACK LITERALLY
56	0000H	1	CSPEC. BYTE EXTERNAL(32)
56	0000H	2	CURCSEG. WORD EXTERNAL(28)
56	0000H	2	CURDSEG. WORD EXTERNAL(29)
56	0000H	2	CURESEG. WORD EXTERNAL(31)
56	0000H	9	CURRENTSYMBOL. STRUCTURE EXTERNAL(38)
56	0000H	2	CURSSEG. WORD EXTERNAL(30)
333	036EH	11	DBCMROUT PROCEDURE PUBLIC STACK=0016H
121	0092H	108	DBDMCOMMONROUT PROCEDURE STACK=000CH 290 291
362	03C6H	11	DBITCMROUT PROCEDURE PUBLIC STACK=0016H
210	019EH	11	DBITROUT PROCEDURE STACK=0006H 299

COPYRIGHT © 1981, 1982

DIGITAL RESEARCH

P.O. BOX 579

FARMER GROVE, CA 95950

SER. #

2		DCL.	LITERALLY				
344	0384H	11 DDCMROUT	PROCEDURE PUBLIC STACK=0016H				
56	0000H	1 DEFAULTDRIVE	BYTE EXTERNAL(75)				
3		DELETEDSYMB.	LITERALLY				
56	0001H	1 DESCR.	BYTE MEMBER(TOKEN)				
121	0004H	1 DIRECTIVE.	BYTE PARAMETER AUTOMATIC	122	125	126	143
159	0004H	1 DIRECTIVE.	BYTE PARAMETER AUTOMATIC	160	165		
3		DOUBLEDEFINED.	LITERALLY				
17		DOUBLEDEFLAB	LITERALLY				
17		DOUBLEDEFSYMB.	LITERALLY				
17		DOUBLEDEFVAR	LITERALLY				
16		DOUBLERANGE.	LITERALLY				
56	0000H	2 DSCIP.	WORD EXTERNAL(27)				
12		DSOVER	LITERALLY				
56	0000H	1 DSPEC.	BYTE EXTERNAL(33)				
137	00FEH	56 DSRBRWROUT	PROCEDURE STACK=000CH	292	293	296	297
341	0379H	11 DWCMROUT	PROCEDURE PUBLIC STACK=0016H				
4		DWRD	LITERALLY				
2		ELSEDO	LITERALLY				
30	0000H	2 EXPTYLINE.	PROCEDURE BYTE EXTERNAL(5) STACK=0000H				
2		ENDCASE.	LITERALLY				
365	03D1H	83 ENDCMROUT.	PROCEDURE PUBLIC STACK=000AH				
271	0270H	11 ENDDBITROUT.	PROCEDURE STACK=0006H.	301			
2		ENDDO.	LITERALLY				
2		ENDFOREVER	LITERALLY				
2		ENDIF.	LITERALLY				
2		ENDMODULE.	LITERALLY				
274	027BH	11 ENDMROUT	PROCEDURE STACK=0006H	377			
2		ENDOFFILE.	LITERALLY				
17		ENDOFFLINEERR	LITERALLY				
2		ENDPROC.	LITERALLY				
2		ENDWHILE	LITERALLY				
56	0000H	1 EOFSET	BYTE EXTERNAL(59)				
32	0000H	ERRMSG	PROCEDURE EXTERNAL(6) STACK=0000H	307	334	382	
32	0000H	1 EKRNU.	BYTE PARAMETER 33				
3		ERROR.	LITERALLY				
56	0000H	1 ERRORPRINTED	BYTE EXTERNAL(62)				
56	0000H	2 ERRORS	WORD EXTERNAL(52)				
56	0000H	2 ESCIP.	WORD EXTERNAL(24)				
12		ESOVER	LITERALLY				
56	0000H	1 ESPEC.	BYTE EXTERNAL(35)				
2		FALSE.	LITERALLY	62	284	318	319 369 372 385 386
213	01A9H	199 FIELDDESCROUT	PROCEDURE STACK=000AH	300			
8		FILEISTRUCTURE	LITERALLY				
8		FILEOSTRUCTURE	LITERALLY				
17		FILESYNTErr.	LITERALLY				
53		FIRST.	LITERALLY				
17		FIRSTITEM.	LITERALLY				
56	0000H	2 FIRSTMACROPTR.	WORD EXTERNAL(63)				
2		FOREVER.	LITERALLY	215			
40	0000H	FORMALLISTROUT	PROCEDURE EXTERNAL(13) STACK=0000H	330			
2		FORMFEED	LITERALLY				
56	0000H	1 FULLSYMBTAB.	BYTE EXTERNAL(65)				
35	0000H	1 GLOBALCMERROR.	BYTE EXTERNAL(8)	306	319	333	385
57	0000H	GLOBALINIT	PROCEDURE EXTERNAL(78) STACK=0000H				
56	0000H	5 HELP	BYTE ARRAY(5) EXTERNAL(73)				
56	0000H	1 I.	BYTE EXTERNAL(74)				

3		IDENT.	LITERALLY	79	103	139	161	230	
56	0000H	1 IFLEVEL.	BYTE EXTERNAL(57)						
56	0000H	1 IFLIST.	BYTE EXTERNAL(38)						
17		IFPARERR.	LITERALLY						
17		ILLEGALMACRO. . . .	LITERALLY						
17		ILLEGALPSEUDO. . .	LITERALLY						
17		ILLEXPRELEM. . . .	LITERALLY						
17		ILLIOPER.	LITERALLY						
56	0000H	1 INCLUDEDDEFAULT. .	BYTE EXTERNAL(76)						
56	0000H	1 INCLUDEON.	BYTE EXTERNAL(56)						
8		INDEXREGBIT. . . .	LITERALLY						
8		INDEXREGCOUNT. . .	LITERALLY						
2		INIT.	LITERALLY						
36	0000H	INITCHROUT.	PROCEDURE EXTERNAL(11)	STACK=0000H				324	
17		INSTRERR.	LITERALLY						
56	0000H	1 INTELHEXON.	BYTE EXTERNAL(50)						
8		IREGBIT.	LITERALLY						
8		IREGCOUNT.	LITERALLY						
3		LAB.	LITERALLY						
17		LABOUTOFRANGE. . .	LITERALLY						
7		LEFTBRACKET. . . .	LITERALLY						
55		LEFTPAR.	LITERALLY	224	242				
44	0000H	LEGALPARAMETER. . .	PROCEDURE BYTE EXTERNAL(15)	STACK=0000H				81	105
			232					141	163
								200	
47	0000H	LEGALSEGREG. . . .	PROCEDURE BYTE EXTERNAL(16)	STACK=0000H				183	
56	0000H	2 LENGTH.	WORD MEMBER(CURRENTSYMBOL)						
56	0000H	2 LENGTH.	WORD MEMBER(OPERANDS)						
2		LF.	LITERALLY						
44	0000H	1 LG.	BYTE PARAMETER	45					
2		LIT.	LITERALLY	2	3	4	5	6	7
				13	14	15	16	17	55
56	0000H	2 MACROPTR.	WORD EXTERNAL(54)						
56	0000H	1 MAXCOL.	BYTE EXTERNAL(39)						
9		MDBF.	LITERALLY						
9		MDBIT.	LITERALLY	211					
9		MDBN.	LITERALLY	270					
9		MDDF.	LITERALLY	292					
9		MDDF.	LITERALLY						
9		MDDN.	LITERALLY	126	291				
9		MENDBIT.	LITERALLY	272					
9		MENDM.	LITERALLY	275					
9		MFORMALBITS. . . .	LITERALLY	263					
17		MISSIFERR.	LITERALLY						
17		MISSINGPSEUDO. . .	LITERALLY						
17		MISSINSTR.	LITERALLY						
17		MISSSEGMINFO. . . .	LITERALLY						
17		MISSTYPEINFO. . . .	LITERALLY						
9		MMODRM1.	LITERALLY	68					
9		MMODRM2.	LITERALLY	83					
9		MNOSEGFIX.	LITERALLY	182					
9		MNUMBERBITS. . . .	LITERALLY	221					
11		MOD8.	LITERALLY						
11		MODD.	LITERALLY						
15		MODLETTERBIT. . . .	LITERALLY						
15		MODLETTERCOUNT. .	LITERALLY						
353	03A5H	11 MODRCHROUT. . . .	PROCEDURE PUBLIC	STACK=0016H					
64	0012H	128 MODRHROUT. . . .	PROCEDURE	STACK=000AH				295	

COPYRIGHT © 1981, 1982

DIGITAL RESEARCH

P. O. BOX 579

PACIFIC GROVE, CA 93950

SER. # _____

11		MODSB.	LITERALLY	
11		MODW.	LITERALLY	
59	0000H	18 MORELEFTONLINE. . .	PROCEDURE BYTE STACK=0002H	304 331 380
9		HRELB.	LITERALLY	296
9		HRELB.	LITERALLY	297
9		HSEGFIX.	LITERALLY	293
38	0000H	NAMEROUT.	PROCEDURE BYTE EXTERNAL(12) STACK=0000H	326
3		NEGLECTED.	LITERALLY	
17		NEGLECTERR.	LITERALLY	
17		NESTEDIFERR.	LITERALLY	
17		NESTEDINCLUDEERR. . .	LITERALLY	
56	0000H	1 NEXTCH.	BYTE EXTERNAL(54)	
4		NIL.	LITERALLY	
56	0000H	1 NOERROR.	BYTE EXTERNAL(61)	
11		NOMOD.	LITERALLY	
56	0000H	1 NOOPER.	BYTE EXTERNAL(70)	
8		NDOVERCOUNT.	LITERALLY	
8		NDOVERRIDEBIT.	LITERALLY	
214	0004H	1 NOPAR.	BYTE	232 240
181	0003H	1 NOPAR.	BYTE	200 202
160	0002H	1 NOPAR.	BYTE	163 166
122	0001H	1 NOPAR.	BYTE	141 144
65	0000H	1 NOPAR.	BYTE	81 84 105 107
13		NOPARAND.	LITERALLY	
16		NORANGE.	LITERALLY	
350	039AH	11 NOSEGFIXCMROUT. . .	PROCEDURE PUBLIC STACK=0016H	
180	0136H	104 NOSEGFIXROUT. . .	PROCEDURE STACK=000AH	294
3		NUMBER.	LITERALLY	66 123 216 249
16		NUMBERRANGE.	LITERALLY	
7		OAND.	LITERALLY	
7		OEQ.	LITERALLY	
56	0006H	2 OFFSET.	WORD MEMBER(OPERANDS)	
56	0006H	2 OFFSET.	WORD MEMBER(CURRENTSYMBOL)	
7		OGE.	LITERALLY	
7		OGT.	LITERALLY	
7		OLAST.	LITERALLY	
7		OLE.	LITERALLY	
7		OLENGTH.	LITERALLY	
7		OLT.	LITERALLY	
7		OMOD.	LITERALLY	
7		ONE.	LITERALLY	
7		ONOT.	LITERALLY	
7		OOFFSET.	LITERALLY	
7		OOR.	LITERALLY	
56	0000H	36 OPERANDS.	STRUCTURE ARRAY(4) EXTERNAL(71)	
8		OPERANDSTRUC.	LITERALLY	56
3		OPERATOR.	LITERALLY	
17		OPNISHATCH.	LITERALLY	
7		OPTR.	LITERALLY	
7		OSEG.	LITERALLY	
7		OSHL.	LITERALLY	
7		OSHORT.	LITERALLY	
7		OSHR.	LITERALLY	
7		OTYPE.	LITERALLY	
7		OXOR.	LITERALLY	
56	0000H	1 PAGESIZE.	BYTE EXTERNAL(45)	
56	0000H	1 PASS.	BYTE EXTERNAL(19)	279 312 366 378

COPYRIGHT © 1981, 1982
 DIGITAL RESEARCH
 P.O. BOX 578
 PACIFIC GROVE, CA 93950
 SER. =

5		RAX.	LITERALLY	
5		RBH.	LITERALLY	
5		RBL.	LITERALLY	
5		RBP.	LITERALLY	
5		RBX.	LITERALLY	
5		RCH.	LITERALLY	
5		RCL.	LITERALLY	
5		RCS.	LITERALLY	189
5		RCX.	LITERALLY	
5		RDH.	LITERALLY	
5		RDI.	LITERALLY	
5		RDL.	LITERALLY	
5		RDS.	LITERALLY	191
5		RDX.	LITERALLY	
?		REENT.	LITERALLY	
3		REG.	LITERALLY	
16		REGISTERRANGE. . .	LITERALLY	
356	03B0H	11 RELBCMR0UT . . .	PROCEDURE PUBLIC STACK=0016H	
359	03B8H	11 RELWCHR0UT . . .	PROCEDURE PUBLIC STACK=0016H	
5		RES.	LITERALLY	188
7		RIGHTBRACKET . . .	LITERALLY	
55		RIGHTPAR	LITERALLY	256 263
5		RSI.	LITERALLY	
5		RSP.	LITERALLY	
5		RSS.	LITERALLY	190
56	0000H	12 SAVESOURCE . . .	BYTE ARRAY(12) EXTERNAL(41)	
20	0000H	SCAN	PROCEDURE EXTERNAL(1) STACK=0000H	96 102 108 135 145 167
			193 199 203 223 229 241 247 255 261 264 268	
18	0000H	SCANINIT.	PROCEDURE EXTERNAL(0) STACK=0000H	
55		SECOND	LITERALLY	
347	03BFH	11 SEGFIXCHR0UT . . .	PROCEDURE PUBLIC STACK=0016H	
8		SEGBIT.	LITERALLY	
8		SEGMCOUNT.	LITERALLY	
56	0004H	2 SEGMENT.	WORD MEMBER(OPERANDS)	
56	0004H	2 SEGMENT.	WORD MEMBER(CURRENTSYMBOL)	
8		SEGTYPEBIT	LITERALLY	
8		SEGTYPECOUNT . . .	LITERALLY	
56	0003H	1 SFLAG.	BYTE MEMBER(OPERANDS)	
56	0003H	1 SFLAG.	BYTE MEMBER(CURRENTSYMBOL)	
56	0000H	1 SIMFORM.	BYTE EXTERNAL(46)	
16		SINGLERANGE. . . .	LITERALLY	
22	0000H	SKIPRESTOFLINE . .	PROCEDURE EXTERNAL(2) STACK=0000H	281 309 315 336 368 384
27	0000H	SKIPUNTIL.	PROCEDURE BYTE EXTERNAL(4) STACK=0000H	
56	0000H	80 SOURCEBUF. . . .	BYTE ARRAY(80) EXTERNAL(47)	
56	0000H	12 SOURCENAME . . .	BYTE ARRAY(12) EXTERNAL(40)	
56	0000H	1 SOURCEPTR.	BYTE EXTERNAL(48)	
2		SPACE.	LITERALLY	
3		SPEC	LITERALLY	
10		SPECA.	LITERALLY	
10		SPECC.	LITERALLY	
10		SPECD.	LITERALLY	
10		SPECE.	LITERALLY	
24	0000H	SPECIALTOKEN . . .	PROCEDURE BYTE EXTERNAL(3) STACK=0000H	
10		SPECH.	LITERALLY	
10		SPECR.	LITERALLY	
10		SPECS.	LITERALLY	
10		SPECX.	LITERALLY	

COPYRIGHT © 1981, 1982

DIGITAL RESEARCH

P. O. BOX 579

PACIFIC GROVE, CA 93950

SER. # _____

```

56 0000H 2 SSCIP. . . . . WORD EXTERNAL(26)
12 SSOVER . . . . . LITERALLY
56 0000H 1 SSPEC. . . . . BYTE EXTERNAL(34)
3 STRING . . . . . LITERALLY
2 STRUC. . . . . LITERALLY 56
56 0002H 1 STYPE. . . . . BYTE MEMBLR(OPERANDS)
56 0002H 1 STYPE. . . . . BYTE MEMBER(CURRENTSYMBOL)
3 SYMBOL . . . . . LITERALLY
56 0000H 1 SYMBO DEVICE . . . . . BYTE EXTERNAL(43)
8 SYMBOLHEAD . . . . . LITERALLY
8 SYMBOLSTRUC. . . . . LITERALLY 56
56 0000H 2 SYMTABADR . . . . . WORD EXTERNAL(69)
2 TAB. . . . . LITERALLY
42 0000H TERMINATECHROUT. . . . . PROCEDURE EXTERNAL(14) STACK=0000H 379
2 THENDC . . . . . LITERALLY 66 69 79 81 97 103 105 123 126 139 141
161 163 194 200 216 224 230 232 242 249 256 279 285 304
312 320 326 331 366 373 380
56 0000H 30 TITLE. . . . . BYTE ARRAY(30) EXTERNAL(44)
27 0000H 1 TOK. . . . . BYTE PARAMETER 28
24 0000H 1 TOK. . . . . BYTE PARAMETER 25
56 0000H 4 TOKEN. . . . . STRUCTURE EXTERNAL(53) 66 69 75 79 103 123 128 131
133 139 161 216 222 230 249 254
2 TRUE . . . . . LITERALLY 61 71 87 92 99 112 117 132 149 154 171
176 185 196 206 215 218 226 234 244 251 258 287 306 314
322 325 328 333 375
277 0004H 1 TYPE . . . . . BYTE PARAMETER AUTOMATIC 278 289
56 0000H 1 TYPE . . . . . BYTE MEMBER(TOKEN) 66 79 103 123 139 161 216 230 249
8 TYPEBIF. . . . . LITERALLY
8 TYPECOUNT. . . . . LITERALLY
3 UDEFSYMB . . . . . LITERALLY
17 UDEFSYMBOL . . . . . LITERALLY
56 0002H 2 VALUE. . . . . WORD MEMBER(TOKEN) 69 75 128 131 133 216 222 249 254
3 VARIABLE . . . . . LITERALLY
52 0000H 2 W. . . . . WORD PARAMETER 53
4 WRD. . . . . LITERALLY

```

MODULE INFORMATION:

```

CODE AREA SIZE    = 0424H 1060D
CONSTANT AREA SIZE = 0000H 0D
VARIABLE AREA SIZE = 0005H 5D
MAXIMUM STACK SIZE = 0016H 22D
936 LINES READ
0 PROGRAM ERROR(S)

```

END OF PL/M-86 COMPILATION

COPYRIGHT © 1981, 1982

DIGITAL RESEARCH

P. O. BOX 578

PACIFIC GROVE, CA 93950

SER. #

